Engineer

almacenamiento en caché

⏭ guardar respuestas y decidir cuándo pueden reutilizarse

Qué es

El almacenamiento en caché consiste en guardar el resultado de un trabajo, como una respuesta HTTP, el resultado de una consulta o una página generada, y entregarlo a solicitudes posteriores siguiendo reglas que indican cuánto tiempo se puede confiar en él y qué lo retira. La acción es almacenar en caché, y el resultado conservado así está en caché; guardar dentro de un programa los valores devueltos por una función suele llamarse memoización.

El almacén que conserva las copias es la caché. El almacenamiento en caché es la política que la rodea, y esa política responde a tres preguntas. ¿Cuánto tiempo está vigente una copia, es decir, se puede servir sin consultar a nadie? Cuando deja de estar vigente, ¿cómo se comprueba? Y cuando cambia el original, ¿qué elimina o sustituye la copia?

HTTP responde a las tres mediante cabeceras. RFC 9111 considera vigente una respuesta almacenada mientras su antigüedad sea inferior a su período de vigencia, que el origen establece con la directiva `max-age` de `Cache-Control` o con una fecha `Expires`. A partir de ese momento, la respuesta está desactualizada y una caché normalmente debe validarla antes de reutilizarla. Envía una solicitud condicional con un validador que conservó, normalmente el `ETag` de la respuesta en una cabecera `If-None-Match`, y, si no ha cambiado nada, el origen responde 304 Not Modified sin cuerpo. La revalidación sigue costando un viaje de ida y vuelta, pero ahorra la transferencia y el trabajo de volver a construir el cuerpo.

```
HTTP/1.1 200 OK
Cache-Control: max-age=3600, stale-while-revalidate=60
ETag: "v42"
```

Dentro de una aplicación, las mismas preguntas adoptan otras formas. En cache-aside, el código consulta primero la caché y, si no encuentra el dato, lee la base de datos y llena la caché por su cuenta. En write-through, cada escritura actualiza la caché junto con la base de datos, de modo que una lectura encuentra el valor actual. La caducidad de cada entrada, su tiempo de vida o TTL, cumple la función de `max-age`. Incluso una base de datos puede almacenar en caché: una vista materializada de PostgreSQL guarda el resultado de una consulta como una tabla, y ese resultado solo cambia cuando alguien ejecuta `REFRESH MATERIALIZED VIEW`.

Utilice el concepto en una decisión

Aplique «¿Qué es el almacenamiento en caché?» a una decisión real. Podemos ayudarle a relacionar el concepto con el sistema que está construyendo o comprando. Detectar las consultas y los índices que faltan en un sistema cada año más pesado y corregir lo que realmente importa.

Explore nuestros servicios Póngase en contacto →

Compruebe los supuestos detrás del término

Utilice la explicación y los trabajos enlazados para formular preguntas sobre su proyecto. Podemos examinar los supuestos con usted antes de elegir una herramienta. 30 días de datos eléctricos en 180 ms — 180 ms para agregar un mes de lecturas de contadores

30 días de datos eléctricos en 180 ms Póngase en contacto →

Por qué importa

Toda regla de almacenamiento en caché intercambia vigencia por velocidad, y el equilibrio adecuado depende del tipo de datos. Un logotipo puede tener un año. El precio de un producto puede tener un minuto. Un saldo o una cantidad de asientos disponibles deben confirmarse con la fuente antes de que alguien actúe basándose en ellos. Lo más habitual es que el almacenamiento en caché falle cuando se aplica una sola regla a los tres casos.

Los archivos estáticos tienen la respuesta más clara. Cuando un proceso de compilación incorpora al nombre de cada archivo un hash de su contenido, un archivo modificado obtiene una URL nueva, por lo que la URL antigua se puede almacenar en caché durante un año. RFC 8246 añade la directiva `immutable`, que indica al navegador que la respuesta seguirá siendo la misma mientras esté vigente, de modo que incluso una recarga puede omitir la revalidación. La página HTML que nombra esos archivos debe tener una vigencia corta o revalidarse en cada uso; de lo contrario, los visitantes seguirán cargando los nombres antiguos.

Es fácil confundir dos directivas. `no-cache` permite almacenar la respuesta en caché, pero prohíbe reutilizarla sin validarla primero con el origen; `no-store` prohíbe almacenarla por completo. Para una página que cambia con frecuencia,

`no-cache` con un `ETag` ofrece contenido actual y sigue ahorrando la transferencia cuando nada ha cambiado.

La invalidación es la parte difícil. RFC 9111 exige que una caché invalide una URL cuando se complete correctamente una solicitud a ella con un método no seguro, como `POST`, `PUT` o `DELETE`, y el mismo apartado señala que esto solo alcanza a las cachés por las que pasa esa solicitud. Un navegador, una CDN y una caché de aplicación son tres lugares separados, y retirar una copia de uno deja las otras dos intactas. Por eso, la mayoría de los sistemas confían en la caducidad como protección final y utilizan la invalidación explícita como forma de actualizar antes.

Los datos desactualizados también pueden ser útiles. RFC 5861 define `stale-while-revalidate`, que permite servir una respuesta desactualizada durante un número determinado de segundos mientras se obtiene una vigente en segundo plano, y `stale-if-error`, que permite seguir sirviendo una copia desactualizada cuando el origen devuelve un error. Ambas convierten una caducidad estricta en una flexible: el visitante recibe una respuesta de inmediato y la actualización llega después.

Cómo se aplica en la práctica

Muchos usuarios de una plataforma son profesionales del mar y personal de campo que trabajan en el mar, con teléfonos y a menudo sin conexión; para ellos, el almacenamiento en caché hizo que la aplicación web se pudiera instalar y utilizar en el mar. El service worker se configura mediante next-pwa, y el almacenamiento en caché durante la ejecución de Workbox sirve fuentes, imágenes, audio, vídeo y recursos de CDN dando prioridad a la caché: desde el dispositivo cuando hay una copia, y desde la red solo cuando no la hay. Son archivos que permanecen iguales de una solicitud a otra y van acompañados de cabeceras HTTP de control de caché. Como los service workers necesitan HTTPS, la instalación y el funcionamiento sin conexión se probaron por HTTPS en dispositivos reales. El resultado es una aplicación que se instala, sigue funcionando sin conexión y carga las visitas posteriores desde la caché en lugar de la red. Una regla que da prioridad a la caché omite por completo la comprobación de vigencia mientras exista una entrada, por lo que encaja con archivos cuyo contenido está ligado a su dirección.

En el otro extremo de un sistema hay una base de datos PostgreSQL que pasó a ser diez veces más rápida que contiene usuarios, organizaciones cliente, datos geográficos y lecturas diarias de electricidad. El almacenamiento en caché fue una medida entre varias. Junto a índices específicos, GiST para consultas espaciales y compuestos para filtros por organización cliente y fecha, y al particionamiento temporal de la tabla de electricidad, algunos resultados de consultas se conservaron como vistas materializadas y se configuraron una caché de consultas y un pool de conexiones para reducir la sobrecarga. Los tiempos de respuesta mejoraron 10 veces, las operaciones críticas se ejecutaban en milisegundos y el uso de recursos cayó un 40 %. Esas mejoras corresponden al trabajo en su conjunto, por lo que no se ha medido por separado la aportación del almacenamiento en caché.

Dónde surgen los problemas

Tras un despliegue, algunos visitantes siguen recibiendo la versión antigua. La respuesta no incluía una vigencia explícita, por lo que las cachés estimaron una. RFC 9111 permite utilizar una fracción del tiempo transcurrido desde `Last-Modified` y menciona el 10 % como ajuste habitual, de modo que un archivo modificado por última vez hace un mes puede reutilizarse durante unos tres días sin consultar. Envíe un `Cache-Control` explícito en cada respuesta, incorpore un hash del contenido al nombre de los archivos estáticos y mantenga una vigencia corta para el HTML que apunta a ellos.

Una purga vacía la CDN y los navegadores conservan la copia antigua. Una purga elimina la copia de la CDN y nada más; cada navegador que almacenó la respuesta la conserva hasta que termina su propio período de vigencia. Dé a las respuestas públicas un `max-age` corto para los navegadores y un `s-maxage` más largo, que solo aplican las cachés compartidas, para que la copia que puede purgarse sea la que más tiempo dure. Cuando una copia del navegador deba cambiar inmediatamente, cambie la URL.

Una página que faltó por un momento sigue apareciendo como inexistente. RFC 9110 permite almacenar un 404 en caché mediante una estimación heurística, de modo que una caché puede conservar uno que llegó sin información de vigencia durante el período que ella misma calcule. Un enlace seguido un instante antes de publicar una página puede seguir devolviendo 404 después de que la página exista. Dé a las respuestas de error una vigencia corta explícita y purgue la URL cuando se publique la página.

Se vuelve a escribir un valor antiguo después de una actualización. En cache-aside, un lector no encuentra el dato en caché y lee la fila; antes de que guarde el valor, un escritor actualiza la fila y elimina la entrada; después, el lector guarda lo que leyó y el valor antiguo permanece hasta que algo lo retira. Elimine la entrada después de confirmar la transacción y dé un TTL a cada entrada para que una invalidación afectada por una condición de carrera u omitida termine por resolverse sola. En Redis, un `SET` simple descarta cualquier caducidad anterior, por lo que debe establecer el TTL en el mismo comando:

```
SET price:42 "19.90" EX 300
```

Lecturas adicionales

- RFC 9111, apartado 3: almacenar respuestas en cachés — cuándo se puede almacenar una respuesta, seguido de las reglas de vigencia, revalidación e invalidación.
- RFC 9110, apartado 13: solicitudes condicionales — `ETag`, `If-None-Match` y la respuesta 304 que abaratan la revalidación.
- RFC 5861: extensiones de Cache-Control HTTP para contenido desactualizado — las dos directivas que permiten servir deliberadamente respuestas desactualizadas desde una caché.
- RFC 8246: respuestas HTTP inmutables — la directiva `immutable` para archivos cuya URL cambia con su contenido.
- Workbox: estrategias de almacenamiento en caché — prioridad a la caché, prioridad a la red y stale-while-revalidate en un service worker, con código.
- PostgreSQL: vistas materializadas — conservar el resultado de una consulta dentro de la base de datos y actualizarlo bajo demanda.

Glosario

Elija según sus restricciones reales

Traiga un conjunto de datos representativo, las preguntas que debe responder y el cuello de botella actual. Podemos acordar cómo medir la calidad, la capacidad de procesamiento y la recuperación.

Vea cuánto cuesta Póngase en contacto →

Haga práctica la terminología

Pregunte cómo se aplica «¿Qué es el almacenamiento en caché?» a su organización. Basta un ejemplo concreto de su sistema actual para iniciar la conversación.

Explore nuestros servicios

welcome@engineer.company +45 50 10 61 84 Copenhague, Dinamarca

Llévese esta página

Tarjeta de contacto

Creado con esmero y cariño por D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Derechos de autor © 2025 – actualidad · Engineer ApS · Todos los derechos reservados.

Impreso desde https://engineer.company/es/glossary/caching/